

IPP-HURRAY! Research Group



Polytechnic Institute of Porto
School of Engineering (ISEP-IPP)

Ethernet-based Systems: Contributions to the Holistic Analysis

Nuno PEREIRA
Eduardo TOVAR

HURRAY-TR-0415 (version 01)
April-2004



Ethernet-based Systems: Contributions to the Holistic Analysis

Nuno PEREIRA, Eduardo TOVAR

IPP-HURRAY! Research Group
Polytechnic Institute of Porto (ISEP-IPP)
Rua Dr. António Bernardino de Almeida, 431
4200-072 Porto
Portugal
Tel.: +351.22.8340502, Fax: +351.22.8340509
E-mail: {npereira, emt}@dei.isep.ipp.pt
<http://www.hurray.isep.ipp.pt>

Abstract:

A number of characteristics are boosting the eagerness of extending Ethernet to also cover factory-floor distributed real-time applications. Full-duplex links, non-blocking and priority-based switching, bandwidth availability, just to mention a few, are characteristics upon which that eagerness is building up. But, will Ethernet technologies really manage to replace traditional Fieldbus networks? To this question, Fieldbus fundamentalists often argue that the two technologies are not comparable. In fact, Ethernet technology, by itself, does not include features above the lower layers of the OSI communication model. Where are the higher layers that permit building real industrial applications? And, taking for free that they are available, what is the impact of those protocols, mechanisms and application models on the overall performance of Ethernet-based distributed factory-floor applications? In this paper we provide some contributions that may pave the way towards providing some reasonable answers to these issues.

Ethernet-based Systems: Contributions to the Holistic Analysis

Nuno Pereira, Eduardo Tovar

Polytechnic Institute of Porto, Porto, Portugal
{npereira, emt}@dei.isep.ipp.pt

Abstract

A number of characteristics are boosting the eagerness of extending Ethernet to also cover factory-floor distributed real-time applications. Full-duplex links, non-blocking and priority-based switching, bandwidth availability, just to mention a few, are characteristics upon which that eagerness is building up. But, will Ethernet technologies really manage to replace traditional Fieldbus networks? To this question, Fieldbus fundamentalists often argue that the two technologies are not comparable. In fact, Ethernet technology, by itself, does not include features above the lower layers of the OSI communication model. Where are the higher layers that permit building real industrial applications? And, taking for free that they are available, what is the impact of those protocols, mechanisms and application models on the overall performance of Ethernet-based distributed factory-floor applications? In this paper we provide some contributions that may pave the way towards providing some reasonable answers to these issues.

1. Introduction

Arguments against the use of Ethernet in industrial environments have almost disappeared. “Familiarity”, “high availability” (subsequently, low cost) and improved timeliness and dependability are driving this phenomenon. But still, there are obstacles to overcome [1]. Indeed, recent research efforts [2, 3] on Ethernet technologies have been focusing on timeliness, trying to find solutions to issues such as bounded response time evaluation, optimal scheduling policies, switching topologies or clock synchronisation. However, they essentially consider the timing characteristics at the Data Link Layer, and it is still to come, to our best knowledge, an overall approach embracing a fully defined protocol stack. While until a couple of years ago a valid justification for this hole could eventually be the actual lack of technologies [4] offering an overall ensemble of protocols and mechanisms, this justification can not serve that purpose anymore.

Ethernet/IP [5], where IP stands for “Industrial Protocol”, is eventually just only one example of a commercial-off-the-shelf (COTS) technology offering a full set of protocols and mechanisms enabling the development of distributed time-critical applications for the factory-floor environment. Ethernet/IP uses an Application protocol – the Control and Information Protocol (CIP), layered on top of a standard TCP/IP protocol stack, where the physical and data link layers can be commodity Ethernet technologies.

In this paper, we contribute a first approach to the worst-case end-to-end latency evaluation of distributed real-time systems based on Ethernet/IP.

2. Basics of Ethernet/IP-like Networks

In this section, we will describe the main characteristics of Ethernet/IP ensembles. We denote those as Ethernet/IP-like technologies, to stress the fact that some of the architectural details and implementations are open to alternative options from technology providers.

Essentially, Ethernet/IP defines a protocol stack that uses CIP as Application protocol. CIP is layered on top of a standard TCP/IP protocol stack, where the physical medium is commodity Ethernet. CIP uses an abstract object modelling approach to describe the suite of communication services available, the externally visible behaviour of a CIP node and a common means by which information within CIP products is accessed and exchanged. The majority of the messaging performed on a CIP Network is done through connections. CIP connections define the packets that will be produced on the network, and can be of two types: Explicit Messaging or Implicit Messaging.

Implicit messaging is the messaging used for time critical I/O data, and therefore will receive the focus of our attention, specially the Cyclic Implicit CIP type of connections. A device produces cyclic messages on a predetermined rate basis, defined by the Requested Packet Interval (RPI) parameter. Underlying these transactions is a producer/distributor/consumer model, also usually found in other factory communication networks [6]. In Ethernet/IP networks the distribution is supported upon multicast UDP/IP that, in turn, is mapped onto the multicast service of Ethernet.

Ethernet/IP-like networks are constituted of three structuring types of nodes: Remote I/Os, Controllers and interconnecting Switches. These nodes communicate with each other via Ethernet. Diverse modules can compose the Remote I/O and Controller nodes. These modules communicate among them via a device-specific backplane (Figure 1). Typically, a Controller is composed of a number of I/O modules (labelled in the figure as *I* or *O*), several controller modules (*C*) and one or more Ethernet Adapters (*EA*). A Remote I/O node has no Controller modules.

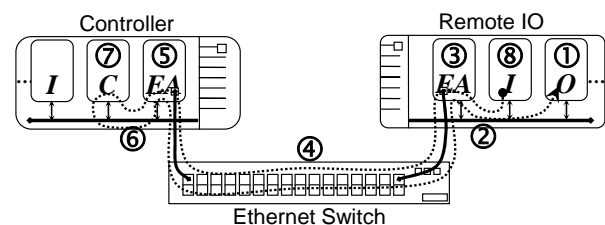


Figure 1. Ethernet/IP-like networks basic nodes and an end-to-end transaction example

Before closing this section it is worth to take a closer look to the type of end-to-end transactions we are addressing. Assume a simple network scenario (Figure 1), and an end-to-end transaction between the Remote I/O and the Controller nodes.

The above mentioned transaction starts at the input module of the Remote I/O (⑧), where a message with the actual input data will be generated (produced) at a rate defined by the RPI parameter for that particular connection. This message will suffer contention delay at the node device backplane (②), and will then arrive at the Ethernet Adapter, where it is processed, and sent via the Ethernet communication interface (③). The Ethernet switch forwards the message to the corresponding output port(s) (④) with a determinable latency. The message will arrive to the Controller Ethernet Adapter (⑤), where is parsed and dispatched to the Controller module via the node backplane (⑥). At the controller (consumer of the data related to the transaction), the input data will be processed by a controller task (characterised by a worst-case response time), that generates the corresponding output data (⑦). The generated output data corresponds to another transaction, in this case produced by the controller and consumed at the Remote I/O node. With another RPI parameter associated, this message will then follow the inverse path (⑥,⑤,④), until it reaching the EA of the Remote I/O (③). It is then processed and delivered to the output module that will, in result, energise the corresponding output(s) (①).

3. A Formulation for the End-to-end Latency

There is a trend towards approaching the problem of engineering distributed real-time systems with alternatives to the usual framework dominated by the notion of absolute temporal guarantees. One of the main arguments is that distributed systems tend to be more flexible and adaptive in their nature. In this direction, a great amount of research [7] is being performed towards including, into the traditional analytical models for computing worst-case response time, some stochastic representation of the events. Clearly, this may be useful if the application can cope with occasional deadline misses, within some quantifiable limits. This could potentially be a reasonable framework for some particular Ethernet/IP-based distributed applications.

Also it is known that for many large-scale distributed systems, analytical-based worst-case formulations tend to be overwhelmed with simplifications that often lead to pessimistic assumptions, and therefore to very pessimistic worst-case results. Of course, a number of techniques exist that can be used and adapted to reduce this pessimism level. However the benefit may appear at the cost of adding rather complex abstractions, such as precedence relationships, event phasing and inheritance of time characteristics. These, unfortunately, may lead to intractable mathematical models, thus making it further difficult to handle and reason the analytical abstractions.

After this short discussion, it is important to stress that the Producer/Distributor/Consumer model underlying Ethernet/IP transactions brings the advantage of de-coupling the diverse latency components. This is an asset that supports tackling the problem in a guaranteed worst-case fashion. We are however

aware that other sources of concern can nevertheless emerge from the non-synchronism between distribution rates and actual reading/actuation of the controlled-environment variables. These concerns are not tackled in this paper.

3.1 Overall Formulation

Therefore, and considering the type of transactions described in the previous section, one can formulate the end-to-end latency as follows:

$$R_i = fd_i + \sum_{m \in \{input, output\}} (Li_m^{sn \rightarrow sw} + Li_m^{sw} + Li_m^{sw \rightarrow dn}) + R\tau_i \quad (1)$$

In brief, the delay associated to an end-to-end transaction results from the delay components associated to two independent transactions (the \sum term), added to the worst-case controller task response time ($R\tau_i$) and the input filter delay (fd_i), (for accounting for the time needed to energise the input/output pin [3]).

In equation (1) $Li_m^{sn \rightarrow sw}$ denotes the worst-case time that a message m takes to arrive from a source node sn to a switch sw . In the case $m = input$, sn is considered to be the node that contains the input module related to the overall transaction i . In the case $m = output$, then sn is considered to be the node that includes the controller responsible for processing the output related to overall transaction i . Therefore, $Li_m^{sn \rightarrow sw}$ can be defined as follows:

$$Li_m^{sn \rightarrow sw} = \begin{cases} T_{iRPIinput} + Qb_m^{sn} + Qea_m^{sn}, & \text{if } m = input \\ Qb_m^{sn} + Qea_m^{sn}, & \text{if } m = output \end{cases} \quad (2)$$

where $T_{iRPIinput}$ denotes the time span corresponding to the periodicity defined for the message m connection (*input RPI*) related to overall transaction i , Qb_m^{sn} denotes the worst-case delay caused by access contention in node sn backplane, and Qea_m^{sn} denotes the worst-case delay for message dispatching at the Ethernet adapter of node sn .

Similarly, $Li_m^{sw \rightarrow dn}$ corresponds to the worst-case delay a message m may experience from the switch sw to the destination node dn . In the case $m = input$, dn is considered to be the node that contains the controller module related to the overall transaction i . In the case $m = output$, dn is considered to be the node that includes the output module related to overall transaction i . Therefore, $Li_m^{sw \rightarrow dn}$ is defined as:

$$Li_m^{sw \rightarrow dn} = \begin{cases} Qb_m^{dn} + Qea_m^{dn}, & \text{if } m = input \\ T_{iRPIoutput} + Qb_m^{dn} + Qea_m^{dn}, & \text{if } m = output \end{cases} \quad (3)$$

where $T_{iRPIoutput}$ denotes the time span corresponding to the periodicity defined for the message m connection (*input RPI*) related to overall transaction i .

In (1), Li_m^{sw} denotes the worst-case relaying delay a message may experience at Ethernet switch sw . This latency includes the time taken by the switch to relay message m to the corresponding output port, and the queuing delay the message may suffer at the output port. This latency will be reasoned out later on in a separate sub-section.

Finally, and before going through further details, a few words on the computation of $R\tau_i$. Typical Ethernet/IP controller modules support fixed priority scheduling.

Therefore, it is possible to obtain the worst-case response-time for the task associated to overall transaction i ($R\tau_i$) by applying well-known response time analysis.

3.2 Latency Introduced by the EA (Qea_m)

For the sake of simplicity, a rough characterisation is adopted for analysing the latency introduced by the Ethernet Adapters (EA). We assume that messages are handled in an on-demand fashion: as soon as a packet fully arrives at the network interface, a “packet arrived” interrupt is raised on the host processor. The interrupt handler releases a task which copies the data from the network buffer, performs the necessary delivery operations to a task that in turn will encapsulate the data and transmit it to the remote Ethernet address. Some delay components (Figure 2) are considered such as the delivery delay of the message (④), the generation delay of the encapsulated message (⑤), the possible queuing to deliver it to the Ethernet network interface (⑥), and, finally, a transmission and a propagation delay (⑦). We will denote the worst-case aggregating all these latencies as D^{ea} ; that is, the worst-case processing delay for any message being processed at the particular Ethernet adapter of node ea . Note that ea will correspond to sn or dn , depending on the formulations under consideration (equations (2) or (3)). Therefore, a simple worst-case formulation of the delay introduced by the EA considers that a particular message m will be processed only after all possible contending messages (nc_m^{ea}) are processed:

$$Qea_m^{ea} = nc_m^{ea} \times D^{ea}, \text{ with } ea \in \{sn, dn\} \quad (4)$$

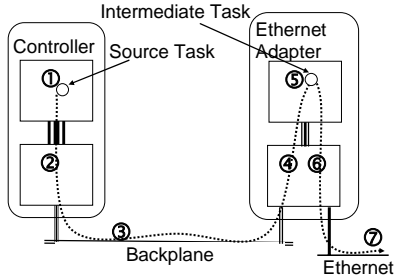


Figure 2. Controller: Message delay components

3.3 Latency Introduced by the Backplane (Qb_m)

Figure 2 also illustrates other components contributing to the overall worst-case latencies. The generation delay introduced by the task processing the related output object to execute and generate the packet (①) (this corresponds to $R\tau_i$). The access delay, when sending the message to the backplane of the Controller (②) (Qb_m) and the propagation delay in the backplane (③). We neglect this latter, and will now reason about the second one.

We assume the backplane to have a medium access control schema based on a Time Division Multiple Access (TDMA) protocol to grant access to it to the diverse modules making the Ethernet/IP node. We assume that each TDMA cycle, each module is able to transmit only one message. Each slot as a fixed duration larger than the time needed to transmit the largest message transferred in the node’s backplane. Therefore, assuming that ts is the duration of a backplane

TDMA slot, in one Ethernet/IP node the TDMA cycle time duration is a function of ts and the number of modules (nm^{nod}) associated to it. Let $V_{nod} = ts \times nm^{nod}$ be the TDMA cycle time duration of Ethernet/IP node nod .

Messages generated in one module contending for backplane access are assumed to be scheduled according to a non pre-emptive rate monotonic policy (priorities based on the periodicities of the connections – the RPIs). The traditional worst-case response time for non pre-emptive tasks in uniprocessor systems can easily be adapted to encompass the worst-case message access delay to device backplanes (Qb_m) as follows. The worst-case queuing time that a message m can experience to access the backplane (Qb_m) is given by the sum of the interference caused by higher priority messages (I_m) and the message transmission (C_m) (we consider negligible the propagation delay in the backplane):

$$Qb_m = I_m + C_m \quad (5)$$

Using the before mentioned analogy, the worst-case interference time a message can suffer to be transmitted to the backplane is given by:

$$I_m = B_m + \sum_{j \in hp(m)} \left\lceil \frac{I_m}{T_j} \right\rceil \times V = V \times \left(1 + \sum_{j \in hp(m)} \left\lceil \frac{I_m}{T_j} \right\rceil \right) \quad (6)$$

B_m is the blocking time that a message may experience due the arrival of a lower priority message an instant ahead of message m . This is the maximum time taken to transfer a lower priority message ($B_m = V$). V is always the interference resulting when a message is scheduled ahead. In (6), $hp(m)$ denotes the set of messages with higher priority than m , and T_j the respective periodicity (inherited from the RPIs). Trivially, I_m can be found by forming a recurrence relationship.

3.4 Latency Introduced by the Switch (L^{sw})

Most modern Ethernet switches support full duplex operation, allowing simultaneous two-way transmission over point-to-point links. Since switches provide a separate collision domain for each port, using full-duplex communication, collisions are avoided. Recent switches typically announce wire-speed and non-blocking operation, meaning that all ports can simultaneously transmit or receive at their full data rates. Furthermore, with IEEE 802.1p, modern switches may have the capability to prioritise messages and, with 802.1q, it is possible to create virtual LANs (VLANs), providing traffic isolation between logically separated networks.

In a preliminary analysis we will contemplate a switch that implements priorities, based on classification of the Ethernet frames. We will disregard the option of isolating traffic with VLANs and consider that, under a controlled load the switch will introduce constant switching delay. If traffic is sent to an output port at a higher rate than its capacity, packets must be queued. A relevant problem arises from the fact that typical implementations of Ethernet priorities support a reduced number of priorities, often mandating that the message priorities have to be mapped into the number of priorities

supported by the switch. Assuming that the original priorities are aggregated into groups of priorities, the following formulation may give the worst-case queuing time in a switch with a first-come-first-served policy between messages of equal priority.

$$L_m^{sw} = I_{s_m} + C_{s_m} + \sum_{j \in cp(m)} C_{s_j} \quad (7)$$

where, $cp(m)$ is the set of messages from connections going out through the same switch port as message m . C_{s_m} is the time to transmit a message m , including the inter-frame delay. I_{s_m} is defined by the sum of the maximum blocking time a message may experience, including the blocking by messages of equal priority and the interference from higher priority messages:

$$I_{s_m} = Dsl \times (1 + neq(m)) + \sum_{j \in hp(m)} \left\lceil \frac{I_m}{T_j} \right\rceil \times Dsl \quad (8)$$

In (8), $neq(m)$ is the number of messages with priority equal to m , and Dsl includes the latency introduced by the switch to classify and relay the frame to an output port. More sophisticated formulations have been tackled previously which may be considered into this analysis [2, 3]. The formulations we used are just for the sake of simplicity.

4. Numerical Example

For the purpose of instantiating the formulation presented, a scenario with eight end-to-end transactions, depicted in Figure 3, was setup.

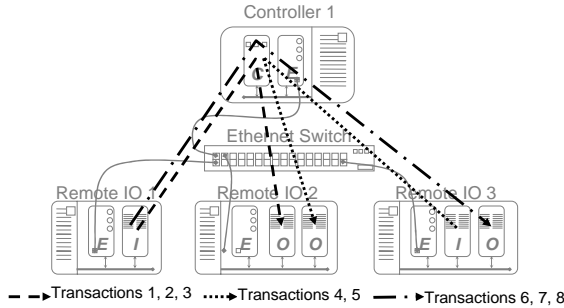


Figure 3. Example scenario

The Controller module has two tasks that process their input data. Task τ_1 , with a worst-case response time of 2 ms processes input data from the first five connections (1,2,3,4,5), while τ_2 is processing the input data from the last three connections (6,7,8). In order to perform the necessary calculations, some additional, device-specific, parameters are required. The values assumed are included in Table 1, below.

Table 1. Assumptions for device parameters

Description	Value (ms)
Worst-case delay in the EA (Dea)	0.20
Worst-case switching delay (Dsl)	0.011
Inter-frame delay	9.6E-04
Time slot, in the backplane (ts)	0.05

Applying the analytical formulation to this scenario, enable us to reach the worst-case latencies as given in Table 2.

Table 2. Transactions response time results (1)

Tr	Period (ms)	Input (ms)			Output			Ri (ms)
		L_m^{sw-en} (ms)	L_m^{sw} (ms)	L_m^{sw-en} (ms)	L_m^{sw-en} (ms)	L_m^{sw} (ms)	L_m^{sw-en} (ms)	
1	5	6.3	0.05	3.3	8.3	0.04	1.15	21.14
2	7	8.4	0.06	3.4	10.4	0.05	1.3	25.61
3	10	11.5	0.07	3.5	13.5	0.06	1.45	32.08
4	25	26.15	0.08	3.6	28.6	0.07	1.15	61.65
5	30	31.3	0.09	3.7	33.7	0.08	1.3	72.17
6	45	46.3	0.11	3.8	48.8	0.08	1.15	104.24
7	75	76.4	0.12	3.9	78.9	0.09	1.3	164.71
8	150	151.5	0.13	4	154	0.10	1.45	315.18

5. Ongoing Work

In this paper we provide an effort to formulate an analytical solution enabling to find end-to-end response times in Ethernet/IP based distributed systems. To our best knowledge, this is an innovative work, and is part of a larger framework being carried out within our research group. It is known that analytical models to provide real-time guarantees for distributed systems, such as those based on worst-case scenarios presented in this paper, tend to be overwhelmed with simplifications that often lead to pessimistic assumptions, and therefore to pessimistic results. Within the above-mentioned framework, ongoing research [8] is looking at devising discrete event simulation models for Ethernet/IP networks. The purpose is two folded. On one hand, discrete event simulation can be a powerful tool to the actual timeliness evaluation of the overall system. On the other hand, it can provide results enabling less pessimistic assumptions (e.g. on precedence and offsets of events) for the analytical response time, of which this paper is a first approach.

References

- [1] B. Baptista, "Industrial Ethernet: Building Blocks for a Holistic Approach". In Proc. of WIP Session of the 4th IEEE International Workshop on Factory Communication Systems (WFCS'02), 2002.
- [2] Y. Song, A. Koubaa, and F. Simonot, "Switched Ethernet for Real-Time Industrial Communication: Modelling and Message Buffering Delay Evaluation". In Proc. of 4th IEEE Int. Workshop on Factory Communication Systems (WFCS'02), pp. 27-35, 2002.
- [3] J.-P. Georges, E. Rondeau, and T. Divoux, "Evaluation of switched Ethernet in an industrial context by using the Network Calculus". In Proc. of 4th IEEE Int. Workshop on Factory Communication Systems (WFCS'02), pp. 19-26, 2002.
- [4] M. Volz, "Quo Vadis Layer 7". In The Industrial Ethernet Book, vol. no. 5: Spring, 2001, pp. 8-10.
- [5] "Ethernet/IP Specification, Release 1.0", vol. 1-2. ControlNet International and Open DeviceNet Vendor Association, June, 5th, 2001.
- [6] E. Tovar, L. Almeida, J. Fonseca, and F. Vasques, "Schedulability Analysis of Real-Time Traffic in WorldFIP Networks: an Integrated Approach", IEEE Transactions on Industrial Electronics, vol. 49, pp. 1165-1174, 2002.
- [7] J. Díaz, D. García, K. Kim, C. Lee, L. Lo Bello, J. López, S. L. Min, and O. Mirabella, "Stochastic Analysis of Periodic Real-Time Systems". In Proc. of 23rd IEEE Real-Time Systems Symposium (RTSS'02), pp. 289-300, 2002.
- [8] N. Pereira, E. Tovar, and L. M. Pinho, "Timeliness in COTS Factory-Floor Distributed Systems: What Role for Simulation", Polytechnic Institute of Porto, Technical Report. March 2004.